**CIS 246 – Spring 2020**

|  |  |
| --- | --- |
| **Program:** | **6** |
| **Points:** | **20** |
| **Chapter(s):** | **7** |
| **File(s) to Submit:** | **InventoryDriver.cpp, Car.h (as one zip file)** |
| **Due:** | **April 7, 11:59 pm** |

**Summary**

Write a C++ application to handle adding, deleting, and printing cars from a dealer inventory, using the C++ Standard Library class template **vector**.

**Description**

The JJC Car Dealership needs an inventory management system. Your application will display a menu that prompts the user to add or delete cars from inventory, to print all the car information, or to exit the application.

When printing the inventory, display the car makes and prices in a nicely formatted, aligned chart.

***Your code does not have to handle duplicate car types. Test it without using duplicates.***

The menu should continue to redisplay until the user chooses to exit the program.

Use the example code file **fig07\_21.cpp** as a reference for working with vectors.

**Requirements for the Car Class**

private data members:

1. string variable to hold the make of car (Ford, Honda, etc.)
2. integer variable to hold the price

public functions:

1. **Two-argument constructor**
2. **Getter functions** – one for each data member
3. No setter functions are needed for this class.

**Requirements for the InventoryDriver Program**

1. **main**

The main function creates the vector using the Car class, and then calls the displayMenu function to display the menu.

Main must use a **sentinel-controlled loop**. Within the loop, call the displayMenu function, and read the user’s choice from stdin. Still within the loop, use a **selection statement** to call the appropriate add, delete, or print function based on their choice. Then redisplay the menu by calling displayMenu.

Be sure to use the most appropriate type of repetition statement for displaying menus.

1. **displayMenu**

Parameters: none  
Return value: none

This function displays the menu, which should look like this:

**1. Add a car to inventory**

**2. Delete a car from inventory**

**3. Print inventory**

**4. Exit**

**Your choice:**

Note that displayMenu should NOT read the user’s choice. It should **only display**. Main will read user input.

1. **addData**

Parameters: one vector of Car objects  
Return value: none

This function prompts the user for the type and price of the car to be added. Create a new Car object with this information, and then add this this new Car object to the vector by calling the push\_back() function.

1. **deleteData**

Parameters: one vector of Car objects  
Return value: none

This function prompts the user for the type of car to delete. Use a **standard** **counter-controlled** **for loop** **to iterate through the vector** until a match is found. If a match is found, call the erase() function of the vector. If a match is not found, display a warning message.

The erase() function uses the following syntax:

***nameOfVector*.erase(*nameOfVector*.begin() + *the subscript of the Car to be deleted*)**

For example, if the car to be deleted from inventory is at index 6 in the vector, and the vector is called cars, the erase function is called like this:

**cars.erase(cars.begin() + 6);**

1. **printData**

Parameters: one vector of Car objects  
Return value: none

This function uses the **range-based for** **loop syntax to iterate through the vector** and displays each car’s information in a neatly formatted list.

**General Requirements**

For complete credit, you must:

1. **MEET ALL REQUIREMENTS ACCORDING TO THE INSTRUCTIONS** – Follow the instructions as written for completing this project, even if you [think you] know a “better” way to do something.
2. **INCLUDE COMMENTS** – Include comments in your code. There must be a comment at the top of each source code or header file that includes your name, the assignment number, and a description of the code in that file. There must be comments at each important step in your algorithm that describes that step.
3. **FOLLOW BEST PRACTICES** – Follow best practices in C++ programming, including, but not limited to, appropriate use of private/public, appropriate use of classes and/or header files, sets & gets, white space, alignment, meaningful variable names, naming conventions, using statements, etc. Points will be deducted for sloppy code that is hard to read, even if it works, so pay attention to these details.
4. **SUBMIT ALL FILES BEFORE THE DUE DATE** – Submit a .zip of ONLY source code files to the dropbox for this assignment on Canvas before the due date. Do not submit anything except .cpp and/or .h, within a zip. Do not submit .exe files. Do not submit a folder structure. Do not submit project files from an IDE.

**Sample Run**

Welcome to the JJC Car Dealership Inventory Program

1. Add a car to inventory

2. Delete a car from inventory

3. Print inventory

4. Exit

Your choice: 1

Type of car: Ford

Price of car: 21000

1. Add a car to inventory

2. Delete a car from inventory

3. Print inventory

4. Exit

Your choice: 1

Type of car: Honda

Price of car: 23000

1. Add a car to inventory

2. Delete a car from inventory

3. Print inventory

4. Exit

Your choice: 3

Make Price

---- -----

Ford 21,000

Honda 23,000

1. Add a car to inventory

2. Delete a car from inventory

3. Print inventory

4. Exit

Your choice: 2

Type of car to delete: Ford

Ford deleted from inventory

1. Add a car to inventory

2. Delete a car from inventory

3. Print inventory

4. Exit

Your choice: 3

Make Price

---- -----

Honda 23,000

1. Add a car to inventory

2. Delete a car from inventory

3. Print inventory

4. Exit

Your choice: 2

Type of car to delete: VW

No such car in inventory

1. Add a car to inventory

2. Delete a car from inventory

3. Print inventory

4. Exit

Your choice: 4